

## **JP2001325134**

Publication Title:

DIRECTORY SETTING METHOD AND RECORDER

Abstract:

Abstract of JP2001325134

PROBLEM TO BE SOLVED: To set up a name inherent as the name of a directory and suitable for a system and to provide the name so as to be easily understood by a user. SOLUTION: One or plural files generated from a certain application software are recorded in a recording medium as lower files of one directory, application ID inherently allocated to the application software is used as a directory name and the directory name is allowed to correspond to a user presenting name.

Data supplied from the esp@cenet database - Worldwide

-----  
Courtesy of <http://v3.espacenet.com>

(19)日本国特許庁（J P）(12)公 開 特 許 公 報（A）(11)特許出願公開番号  
特開2001－325134  
（P2001－325134A）  
(43)公開日 平成13年11月22日（2001. 11. 22）

(51)Int.Cl.<sup>7</sup>識別記号F Iテームト\*（参考）  
G 0 6 F 12/005 2 0G 0 6 F 12/005 2 0 J 5 B 0 7 5  
17/301 5 017/305 2 0 C 5 B 0 8 2  
1 5 0 A

審査請求 未請求 請求項の数10 O L （全 22 頁）

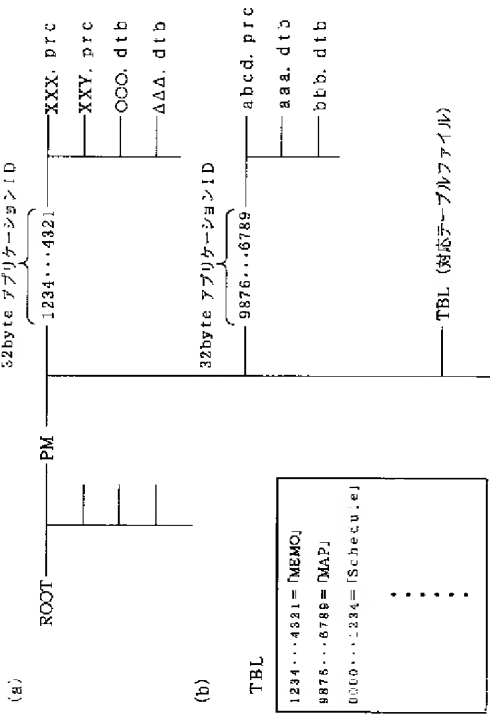
(21)出願番号	特願2000－147479(P2000－147479)	(71)出願人	000002185 ソニー株式会社 東京都品川区北品川 6 丁目 7 番35号
(22)出願日	平成12年 5 月15日(2000. 5. 15)	(72)発明者	松浦 陽子 東京都品川区北品川 6 丁目 7 番35号 ソニ ー株式会社内
		(72)発明者	山崎 友敬 東京都品川区北品川 6 丁目 7 番35号 ソニ ー株式会社内
		(74)代理人	100086841 弁理士 脇 篤夫 Fターム(参考) 5B075 NK10 NK44 NR05 PP03 PQ02 5B082 EA01 EA07

(54)【発明の名称】 ディレクトリ設定方法、記録装置

(57)【要約】

【課題】 ディレクトリの名称として固有かつシステムからみて適切な名称を設定でき、かつユーザーにわかりやすい名称を提示できるようにする。

【解決手段】 或るアプリケーションソフトウェアから発生される 1 又は複数のファイルを、1つのディレクトリの下のファイルとして記録媒体に記録すると共に、アプリケーションソフトウェアに固有に付されているアプリケーションIDをディレクトリ名称として用い、さらに、ディレクトリ名称とユーザ提示用名称の対応づけを行うようにする。



【特許請求の範囲】

【請求項1】 或るアプリケーションソフトウェアから発生される1又は複数のファイルを、1つのディレクトリの下にファイルとして記録媒体に記録すると共に、上記アプリケーションソフトウェアに固有に付されているアプリケーションIDを上記ディレクトリ名称として用い、さらに、上記ディレクトリ名称とユーザ提示用名称の対応づけを行うことを特徴とするディレクトリ設定方法。

【請求項2】 上記ディレクトリ名称と上記ユーザ提示用名称を対応させる対応テーブルファイルを上記記録媒体に記録することで、上記対応づけを実現することを特徴とする請求項1に記載のディレクトリ設定方法。

【請求項3】 上記ユーザ提示用名称を示した名称ファイルを上記ディレクトリ外において、上記記録媒体に記録するとともに、上記名称ファイルを上記ディレクトリにリンクさせることで、上記対応づけを実現することを特徴とする請求項1に記載のディレクトリ設定方法。

【請求項4】 少なくとも上記ユーザ提示用名称を示したタグファイルを、上記ディレクトリ下において、上記記録媒体に記録することで、上記対応づけを実現することを特徴とする請求項1に記載のディレクトリ設定方法。

【請求項5】 上記ディレクトリ下の特定のファイルに、上記ユーザ提示用名称を記録することで、上記対応づけを実現することを特徴とする請求項1に記載のディレクトリ設定方法。

【請求項6】 ユニークなアプリケーションIDを有する或るアプリケーションソフトウェアを構成する上記ユニークなアプリケーションIDを有する1又は複数のファイルを格納するディレクトリを、上記アプリケーションソフトウェアの上記アプリケーションIDをディレクトリ名称として用いて発生させるディレクトリ発生手段と、上記ディレクトリ名称とユーザ提示用名称の対応づけを行う情報を生成する対応情報生成手段と、記録媒体に対して、上記ディレクトリの下にディレクトリ名と同一のIDを有する上記1又は複数のファイルを記録すると共に上記対応情報を記録する記録手段と、を備えたことを特徴とする記録装置。

【請求項7】 上記対応情報生成手段は上記対応情報として、上記ディレクトリ名称と上記ユーザ提示用名称を対応させる対応テーブルファイルを生成し、上記記録手段は、上記対応テーブルファイルを上記記録媒体に記録することを特徴とする請求項6に記載の記録装置。

【請求項8】 上記対応情報生成手段は上記対応情報として、上記ディレクトリにリンクされる、上記ユーザ提示用名称を示した名称ファイルを生成し、上記記録手段は、上記名称ファイルを上記ディレクトリ外において、上記記録媒体に記録することを特徴とする

請求項6に記載の記録装置。

【請求項9】 上記対応情報生成手段は上記対応情報として、少なくとも上記ユーザ提示用名称を示したタグファイルを生成し、

上記記録手段は、上記タグファイルを上記ディレクトリ下において、上記記録媒体に記録することを特徴とする請求項6に記載の記録装置。

【請求項10】 上記対応情報生成手段は上記対応情報として、上記ディレクトリ下の特定のファイルに、上記ユーザ提示用名称を含ませるようにすることを特徴とする請求項6に記載の記録装置。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、情報処理装置で用いるアプリケーションプログラムに関わる情報の記録の際のディレクトリ設定方法、及び記録装置に関するものである。

【0002】

【従来の技術】パーソナルコンピュータやPDA (Personal Digital Assistants: 携帯情報機器) などの情報処理装置においては、HDD (Hard Disc Drive)、光ディスク、光磁気ディスク、磁気ディスク、メモリーカードなどの各種の記録媒体に対するファイル等の記録について、ディレクトリによるツリー構造をとることができる。

【0003】

【発明が解決しようとする課題】ところで、ディレクトリの名称は、他のディレクトリと重複しない固有の名称が要求され、その一方で、例えばユーザーがディレクトリの内容を識別しやすい名称とされることが求められる。さらにシステムからみても内容を判別しやすい名称であることも求められる。例えば情報処理システムにおいて、新たなディレクトリの生成に対して、固有の名称を付すことは、その時点で存在する他のディレクトリの名称と比較確認していくことで容易に実現できる。しかしながら、そのように生成された名称は、必ずしもユーザーにとってわかりやすいものではない。一方、ディレクトリ名称をユーザーが設定するようにすれば、ユーザーにとってはわかりやすいものとなる。しかしながら、名称が重複する可能性はあり、重複した場合は名称変更を余儀なくされる。また、ユーザーに名称入力操作という操作負担を強いることになる。

【0004】

【課題を解決するための手段】本発明はこのような問題点に鑑みて、ディレクトリの名称を、固有かつシステムからみてわかりやすい名称を設定できるようにするとともに、ユーザーにわかりやすい名称を提示できるようにすることを目的とする。

【0005】このために、本発明のディレクトリ設定方法は、ユニーク(固有)なアプリケーションを有する或

るアプリケーションソフトウェアを構成する当該ユニークなアプリケーションIDを有する1又は複数のファイルを、1つのディレクトリの下ファイルとして記録媒体に記録すると共に、上記アプリケーションソフトウェア及びそれを構成するファイル群に固有に付されているアプリケーションIDを上記ディレクトリ名称として用い、さらに、上記ディレクトリ名称とユーザ提示用名称の対応づけを行うようにする。対応づけの手法としては、ディレクトリ名称とユーザ提示用名称を対応させる対応テーブルファイルを記録することや、ユーザ提示用名称を示した名称ファイルをディレクトリにリンクさせること、或いはユーザ提示用名称を示したタグファイルをディレクトリ下に記録すること、さらにはディレクトリ下の特定のファイルに、ユーザ提示用名称を記録すること、などで実現する。

【0006】本発明の記録装置は、或るアプリケーションソフトウェアを構成する1又は複数のファイルを格納するディレクトリを、上記アプリケーションソフトウェア及び構成ファイルに固有に付されているアプリケーションIDをディレクトリ名称として用いて発生させるディレクトリ発生手段と、上記ディレクトリ名称とユーザ提示用名称の対応づけを行う情報を生成する対応情報生成手段と、記録媒体に対して、上記1又は複数のファイルを上記ディレクトリの下に記録すると共に上記対応情報を記録する記録手段と、を備えるようにする。

【0007】

【発明の実施の形態】以下、本発明の実施の形態を次の順序で説明する。なお、実施の形態としては、本発明のディレクトリ設定方法を実行し、また本発明の記録装置に相当する、情報処理装置とする。記録媒体としてはメモ리카ードの例を挙げる。

1. 情報処理装置の外観例
  2. 情報処理装置の構成
  3. OS構造及びデータベース構造
  4. メモ리카ード
    - 4-1 外観
    - 4-2 メモ리카ードの端子及び内部構造
    - 4-3 ファイルシステム処理階層
    - 4-4 物理的データ構造
    - 4-5 物理アドレス及び論理アドレスの概念
    - 4-6 論理-物理アドレス変換テーブル
    - 4-7 ディレクトリ構造
  5. FAT構造
  6. メモ리카ードと情報処理装置のインターフェース
  7. メモ리카ードへのファイル記録時の処理
  8. 形成されるディレクトリ構造例1
  9. 形成されるディレクトリ構造例2
  10. 形成されるディレクトリ構造例3
  11. 形成されるディレクトリ構造例4
- 【0008】1. 情報処理装置の外観例

本例の情報処理装置の外観例を図1に示す。この情報処理装置1は、いわゆるPDA機器として携帯に適した小型軽量の装置とされる。また記録媒体として、後述するメモ리카ード70を装着し、記録再生を行うことができるものとする。なお本発明としては、携帯型の情報処理装置に限られず、パーソナルコンピュータをはじめとするあらゆるタイプの情報処理装置に適用できるものであり、また装置が記録を行う記録媒体はメモ리카ードに限られず、HDD、光ディスク、光磁気ディスク、或いは装置内に固定的に配置されるRAM、フラッシュメモリなど、他の種の記録媒体であってもよいものである。

【0009】図1(a)(b)(c)(d)は情報処理装置1の外観例としての平面図、右側面図、左側面図、上面図を示している。図1(d)に示すように装置上面側には後述するメモ리카ード70を装着可能なメモリスロット7が形成されており、この情報処理装置1は、メモリスロット7に装着されたメモ리카ード70に対する各種データ(コンピュータ用データ、音楽データ、音声データ、動画データ、静止画像データ、制御データなど)の記録再生が可能とされる。なお、この図1の例ではメモリスロット7が2つ形成されていることから、2つのメモ리카ード70を同時に装着できるようになされている。もちろん、形成するメモリスロット7の数は1つでもよいし、3つ以上でもよい。

【0010】この情報処理装置1には、平面上に例えば液晶パネルによる表示部2が形成され、アプリケーションソフトウェアの起動及び各種処理に伴う画像、データとしての画像や文字、再生される音声、音楽に付随する情報、さらには操作のガイドメッセージ、再生や編集操作等のためのメニュー画面などが表示される。

【0011】情報処理装置1上には、ユーザーの操作のための各種の操作子が設けられる。例えば操作キー3a、ジョグダイヤル3b、プッシュダイヤル3cなどがそれぞれ所要部位に形成される。これらの操作子によりユーザーは、例えば電源操作、メニュー操作、選択操作、文字等の入力操作、その他必要とされる各種の操作を行うことができる。これらの操作子はもちろん一例にすぎない。即ち配備する操作子の数、種類、位置は多様に考えられる。

【0012】また、情報処理装置1上には、スピーカ4、マイクロホン5、撮像部6も形成され、音声の出力、入力、撮像による画像の取込なども実行できるようにされている。

【0013】また各種機器との接続のために、各種端子が形成される。例えば図1(b)のように、ヘッドホン端子10、ライン出力端子12、ライン入力端子11などが形成され、また図1(c)のようにIEEE1394端子8、USB(universal serial bus)端子9などが形成される。なお、これらの端子の種類、数、配置位置も、他の例が多様に考えられる。例えば光ケーブル対応

のデジタル入出力端子を備えるようにしたり、或いはS C S Iコネクタ、シリアルポート、R S 2 3 2 Cコネクタなどが形成されるようにしても良い。

#### 【0014】2. 情報処理装置の構成

図2に情報処理装置1の内部構成を示す。図示するように情報処理装置1内には、まず中核となる部位として、システムコントローラ21、CPU22、フラッシュROM23、D-RAM24が設けられる。また基本的なユーザーインターフェースのための部位として操作部35、表示制御部27、表示部2が形成される。

【0015】システムコントローラ21は操作部35からの操作情報を入力し、それに応じてCPU22に割り込みをかける。操作部35とは、図1に示した各種操作子3a、3b、3cに相当する。また図1では説明しなかったが、表示部2に操作キーやアイコンの表示を行うとともに表示部2上でのタッチ検出機構を設けることで、タッチパネル操作子を形成してもよく、その場合のタッチパネル操作子も図2でいう操作部35に含まれるものとなる。

【0016】CPU22は基本ソフト(OS: Operating System)やアプリケーションプログラムが動作される部位となる。CPU22はシステムコントローラ21を介して供給される操作情報に応じて所要の処理を実行する。フラッシュROM23は、基本動作プログラム、各種処理定数、設定情報などを記憶する領域とされる。D-RAM24は、各種処理に必要な情報の記憶、データのバッファリング、CPU22のワークエリアの拡張、その他、CPU22の処理に応じて多様に使用される。またD-RAM24にはストレージエリア(不揮発性領域)が設けられており、そのストレージエリアにはOSやアプリケーションソフトウェアがインストールされる。そしてD-RAM24にインストールされたアプリケーションソフトウェアは、ユーザからの操作に応じて起動され、CPU22により実行される。またアプリケーションソフトウェアはユーザーインターフェース画面を持ち、ユーザーの指示による状態遷移に基づいて、D-RAM24に確保されたフレームバッファに描画を行う。描画された画像データは、表示制御部27に送られ、表示部2に表示される。

【0017】また上述したようにメモリカード70に対するメモリスロット7が形成され、メモリカード70を装着できるが、CPU22は、メモリカードインターフェース28を介して装着されたメモリカード70に対して書込又は読み出しアクセスすることができる。メモリカードインターフェース28とメモリカード70との間のインターフェース動作については後述する。CPU22は、装着されたメモリカード70を、拡張的なメモリ領域として利用することができる。また、もちろんメモリカード70にアプリケーションプログラムが記録されていれば、それをD-RAM24にインストールした

り、或いはアプリケーションやデータを直接D-RAM24に展開して所要処理を実行させることができる。また、或るアプリケーションに基づいてCPU22が、作成した文書データ、画像データ、オーディオデータ、表計算データなどを、メモリカード70に記録することもできる。なお、メモリスロット7にメモリカード70が装着されたことを検出することで、メモリカード70に対する動作が記録再生動作可能になったり、或いはメモリカード70に記録されているアプリケーションやデータが自動的にD-RAM24に展開されるなどの、いわゆるホットプラグイン動作も可能である。またメモリカードインターフェース28は、メモリカード70に記録するデータについての暗号化処理や、読み出したデータの暗号解読処理なども可能とされる。

【0018】撮像部6は例えばCCD撮像素子及び撮像回路系により形成される。撮像部6により取り込まれた撮像画像データは、撮像データインターフェース34を介してD-RAM24に取り込むことができ、またCPU22は所定のアプリケーションプログラムに基づく動作により、撮像画像データの編集やメモリカード70への記録等を実行できる。

【0019】オーディオインターフェース29は、上述したスピーカ4、マイクロホン5、ヘッドホン端子10、ライン出力端子12、ライン入力端子11から入出力されるオーディオデータのインターフェース部位となる。例えばマイクロホン5或いはライン入力端子11から入力されたアナログオーディオ信号は、入力オーディオ処理部32でそれぞれ所定の増幅処理やフィルタリングが行われ、A/D変換器33でデジタルオーディオデータとされてオーディオインターフェース29に供給される。オーディオインターフェース29は、入力されたデジタルオーディオデータについて、CPU22の制御に基づいて処理や出力を実行する。例えば所要の圧縮エンコード処理を行った後、メモリカードインターフェース28に供給し、メモリカード70に記録させることができる。またオーディオインターフェース29は、例えばメモリカード70から読み出されるなどして供給されたデジタルオーディオデータについて所定のデコード処理を行い、D/A変換器30に供給する。D/A変換器30はデジタルオーディオデータをアナログオーディオ信号に変換する。出力オーディオ処理部31は供給されたアナログオーディオ信号について、出力先に応じた所定の増幅処理、インピーダンス調整などを行い、スピーカ4、ヘッドホン端子10、ライン出力端子12に出力する。

【0020】USBインターフェース25は、USBコネクタ9に接続された外部機器との間の通信インターフェースである。CPU22はUSBインターフェース25を介して外部のパーソナルコンピュータ或いは周辺機器などとの間でデータ通信を行うことができる。例えば

この情報処理装置1で扱われる制御データ、コンピュータデータ、画像データ、オーディオデータなどの送受信が実行される。同様にIEEE1394インターフェース26は、IEEE1394端子8に接続された外部機器との間の通信インターフェースである。CPU22はIEEE1394インターフェース26を介して外部の情報機器との間で各種データ通信を行うことができる。

【0021】なお、この図2に示す情報処理装置1の構成はあくまでも一例であり、これに限定されるものではない。即ち、一般にパーソナルコンピュータやPDA機器で採用されている各種構成部位を追加したり、或いは実際の製品として不要の部位を削除することは、設計上の都合により決められるものである。

【0022】3. OS構造及びデータベース構造  
続いて図3で、本例の情報処理装置1に搭載されるOS構造について説明する。図3に示すように、OSは、基本ソフトの中心部分としてのカーネルを含むマネージャ層と、標準ライブラリ、及び制御ICなどのハードウェアのレイヤとなるHAL (Hardware Abstraction Layer) から構成される。アプリケーションソフトウェアは、このようなOS構造による基本動作上で動作される。またHALに対しては、1又は複数のデバイスドライバとして階層が付加され実際のハードウェア(HW)が駆動される。

【0023】ここで、特に本例の情報処理装置1の場合は、メモリカード70をドライブ可能とし、かつ後述するがメモリカード70のデータはFATにより管理されることから、OSにFATライブラリが付加され、さらに、メモリカードをハンドリングするためのライブラリ(MSライブラリ)が付加される。そしてこのFATライブラリ及びMSライブラリに基づいて、メモリドライブがメモリカード70がドライブされる構造とされている。

【0024】このようなOS構造を持つ本例の情報処理装置1では、さらに通常でいうところの「ファイル」に相当する概念として、「データベース」という概念が導入されている。ここでいう「データベース」とは、通常いうところのデータベースのように単にデータを蓄積していったものではなく、データベース自体がデータを管理できる構造としてフォーマット化されている。この意味で、「データベース」は「ファイル」に相当する。

【0025】図4にデータベース構造を示す。即ちデータベースには、ヘッダ(DTBヘッダ)としてデータベースネーム(DTB Name)及びその他情報を含む領域が形成され、さらにポインタテーブルが配される。そしてデータ領域に記録される実際のデータは、ポインタテーブルに記録されたポイント情報により、位置的な管理が行われる状態となっている。

【0026】このような構造のデータベースとしては、2種類のものが存在する。例えば一般に1つのアプリケ

ーションソフトウェアは複数のファイルで構成され、その中には実行ファイル(\*\*\*.exe)と、データファイル(\*\*\*.data)があるが、その実行ファイル(\*\*\*.exe)に相当するものとして「リソースデータベース(\*\*\*.prc)」があり、またデータファイル(\*\*\*.data)に相当するものとして「データベースデータベース(\*\*\*.dtb)」がある。

【0027】本例の情報処理装置1では、このような「データベース」という概念によりデータを扱う。従って、メモリカード70において記録再生されるファイル(FATで扱われるファイル)も、上記データベースの形態となる。なお本明細書では、「ファイル」という言葉を用いるが、これは一般的な概念にあわせて用いているものであり、本実施の形態に関しても、「ファイル」とは上記構造のデータベースの意味となる。

【0028】4. メモリカード

#### 4-1 外観

次にメモリカード70について説明していく。まず図5にメモリカード70の外形状を示す。メモリカード70は、例えば図5に示すような板状の筐体内部に例えば所定容量のメモリ素子を備える。本例としては、このメモリ素子としてフラッシュメモリ(Flash Memory)が用いられるものである。図5に平面図、正面図、側面図、底面図として示す筐体は例えばプラスチックモールドにより形成され、サイズ的具体例としては、図に示す幅W11、W12、W13のそれぞれが、W11=60mm、W12=20mm、W13=2.8mmとなる。

【0029】筐体の正面下部から底面側にかけて例えば10個の電極を持つ端子部72が形成されており、この端子部72から、内部のメモリ素子に対する読出又は書込動作が行われる。筐体の平面方向の左上部は切欠部73とされる。この切欠部73は、このメモリカード70を、例えばドライブ装置本体側の着脱機構へ装填する際に挿入方向を誤ることを防止するためのものとなる。また筐体上面から底面側にかけて、ラベル貼付面74が形成され、ユーザーが記憶内容を書いたラベルを貼付できるようにされている。さらに底面側には、記録内容の誤消去を防止する目的のスライドスイッチ75が形成されている。

【0030】このようなメモリカード70においては、フラッシュメモリ容量としては、4MB(メガバイト)、8MB、16MB、32MB、64MB、128MBの何れかであるものとして規定されている。またデータ記録/再生のためのファイルシステムとして、いわゆるFAT(File Allocation Table)システムが用いられている。

【0031】書込速度は1500KByte/sec~330KByte/sec、読出速度は2.45MByte/secとされ、書込単位は512バイト、消去ブ

ロックサイズは8KB又は16KBとされる。また電源電圧Vccは2.7～3.6V、シリアルクロックSCLKは最高20MHzとされる。

【0032】4-2 メモリカードの端子及び内部構造  
図6に端子部72の電極構造を示す。図5に示したように端子部72は10個の平面電極が1列に並んだ構造とされるが、図6に示すように各電極(端子T1～T10)は次の通りとなる。

【0033】端子T1及びT10は検出電圧Vss端子とされる。端子T2は、シリアルプロトコルバスステート信号BSの入力端子とされる。端子T3及びT9は電源電圧Vcc端子とされる。端子T4はデータ端子、つまりシリアルプロトコルデータ信号の入出力端子とされる。端子T5及びT7はリザーブ(予備)とされる。端子T6は検出端子とされ、ドライブ装置側(情報処理装置1のメモリカードインターフェース)がメモリカードの装着検出に用いる。端子T8は、シリアルクロックSCLKの入力端子とされる。

【0034】また図6にはメモリカード70の内部構成も示している。メモリカード70の内部は、コントロールIC80とフラッシュメモリ81が設けられている。コントロールIC80はフラッシュメモリ81に対する書込/読出動作を実行する部位となる。図からわかるように、コントロールIC80に対しては、端子T2からのシリアルプロトコルバスステート信号BS、端子T8からのシリアルクロックSCLKが供給される。書込動作時には、コントロールIC80は、これらのシリアルプロトコルバスステート信号BS、シリアルクロックSCLKに従って、端子T4から供給されるデータのフラッシュメモリ81への書込を行う。また読出時には、シリアルプロトコルバスステート信号BS、シリアルクロックSCLKに従って、フラッシュメモリ81からデータを読み出し、端子T4からドライブ装置側に出力する。

【0035】また検出電圧Vssは、検出端子T6に供給されており、ドライブ装置側では、図示するように抵抗Rによって検出端子T6の端子電圧を検出することで、このメモリカード70が装着部(メモリスロット7)に接続されているか否かを検出できるようにされる。

【0036】4-3 ファイルシステム処理階層  
続いて、メモリカード70を記録媒体とするシステムにおけるフォーマットについて説明していく。図7は、メモリカード70を記録媒体とするシステムのファイルシステム処理階層を示すものである。この図に示すように、ファイルシステム処理階層としては、アプリケーション処理層の下に、順次、ファイル管理処理層、論理アドレス層、物理アドレス層、フラッシュメモリアクセスがおかれる。この階層では、ファイル管理処理層がいわゆるFAT(File Allocation Table)となる。また、

この図から分かるように、本例のファイルシステムでは論理アドレス及び物理アドレスという概念が導入されているが、これについては後述する。

【0037】4-4 物理的データ構造  
図8には、メモリカード70内の記憶素子である、フラッシュメモリ81の物理的データ構造が示されている。フラッシュメモリ81としての記憶領域は、セグメントという固定長のデータ単位が大元となる。このセグメントは、1セグメントあたり4MB(メガバイト)或いは8MBとして規定されるサイズであり、1つのフラッシュメモリ81内におけるセグメント数は、そのフラッシュメモリ81の容量に依存して異なってくる。

【0038】そして、この1セグメントを図8(a)に示すように、ブロックという固定長のデータ単位として8KB(キロバイト)又は16KBにより区切るようにされる。原則として、1セグメントは512ブロックに区切られることから、図8(a)に示すブロックnについては、 $n=511$ とされることになる。但し、フラッシュメモリ81では、書き込み不可な損傷エリアであるディフェクトエリアとしてのブロック数が所定数の範囲で許可されているため、データ書き込みが有効とされる実質的なブロック数を対象とすれば、上記nは511よりも少なくなる。

【0039】図8(a)に示すようにして形成されるブロック0～nのうち、先頭の2つのブロック0, 1はブートブロックといわれる。但し、実際には有効なブロックの先頭から2つのブロックがブートブロックとして規定されることになっており、必ずしもブートブロックがブロック0, 1である保証はない。そして、残りのブロックが、ユーザデータが格納されるユーザブロックとなる。

【0040】1ブロックは、図8(d)に示すようにして、ページ0～mにより分割される。1ページの容量は、図8(e)に示すように、512バイトのデータエリアと16バイトの冗長部よりなる、528(=512+16)バイトの固定長とされる。なお、冗長部の構造については図8(f)により後述する。また、1ブロック内のページ数としては、1ブロックの容量が8KBの場合には16ページ、16KBの場合には32ページとなる。

【0041】このような、図8(d)(e)に示されるブロック内のページ構造は、上記ブートブロックとユーザブロックとで共通である。また、フラッシュメモリ81では、データの読み出し、及び書き込みはページ単位で行われ、データの消去はブロック単位で行われるものとされる。そして、データの書き込みは、消去済みのページに対してしか行われないものとされている。従って、実際のデータの書き換えや書き込みは、ブロック単位を対象として行われることになる。

【0042】先頭のブートブロックは、図8(b)に示

すように、ページ0に対してヘッダーが格納され、ページ1には初期不良データの位置(アドレス)を示す情報が格納される。また、ページ2にはC I S / I D Iといわれる情報が格納される。2つめのブートブロックは図8(c)に示すように、ブートブロックとしてのバックアップのための領域とされている。

【0043】図8(e)に示された冗長部(16バイト)は、図8(f)に示す構造を有する。この冗長部は、図のように先頭の第0バイト～第2バイトの3バイトが、データエリアのデータ内容の更新に応じて書き換えが可能なオーバーライトエリアとされる。このオーバーライトエリアのうち、第0バイトにはブロックステータスが格納され、第1バイトにはデータステータスが格納される(Block Flag Data)。また、第2バイトの上位の所定ビットを利用して変換テーブルフラグ(Page Data Status1)が格納される。

【0044】原則として第3バイト～第15バイトは、その内容が現ページのデータ内容に応じて固定とされ、書き換えが不可とされる情報が格納される領域となる。第3バイトにはアクセス許可やコピー禁止指定等を示す管理フラグ(Block Info)が格納される。第4、第5バイトから成る2バイトの領域には、後述する論理アドレス(LogicAddress)が格納される。第6～第10バイトの5バイトの領域は、フォーマットリザーブの領域とされ、続く第11、第12バイトの2バイトが、上記フォーマットリザーブに対して誤り訂正を施すための分散情報ECCを格納する領域とされる。残る第13～第15バイトには、図8(e)に示すデータエリアのデータに対して誤り訂正を行うためのデータECCが格納される。

【0045】上記図8(f)に示した冗長部の第3バイトに格納される管理フラグは、図9に示すようにして、ビット7～ビット0の各ビットに、その内容が定義されている。ビット7、6、及びビット1、0はリザーブ(未定義)領域とされている。ビット5は現ブロックに対してのアクセス許可の「有効」(‘1’; Free) / 「無効」(‘0’; Read Protected)を示すフラグが格納される。ビット4には現ブロックについてのコピー禁止指定(‘1’; OK / ‘0’; NG)についてのフラグが格納される。

【0046】ビット3は変換テーブルフラグとされる。この変換テーブルフラグは、現ブロックが後述する論理-物理アドレス変換テーブルであるのか否かを示す識別子であり、このビット3の値が‘0’とされていれば、現ブロックは論理-物理アドレス変換テーブルであることが識別され、‘0’であれば無効となる。つまり、現ブロックは論理-物理アドレス変換テーブルではないことが識別される。

【0047】ビット2はシステムフラグが格納され、‘1’であれば現ブロックがユーザブロックであることが示され、‘0’であればブートブロックであることが

示される。

【0048】ここで、セグメント及びブロックと、フラッシュメモリ容量との関係を図13(左3列を参照)により説明しておく。メモ리카ード70のフラッシュメモリ容量としては、4MB、8MB、16MB、32MB、64MB、128MBの何れかであるものとして規定されている。そして、最も容量の小さい4MBの場合であると、1ブロックは8KBと規定された上で、そのブロック数としては512個とされる。つまり、4MBはちょうど1セグメントの容量を有するものとされる。そして、4MBの容量であれば、同様に1ブロック=8KBの容量が規定された上で、2セグメント=1024ブロックとなる。なお、前述したように、1ブロック=8KBであれば、1ブロック内のページ数は16ページとなる。但し16MBの容量では、1ブロックあたりの容量として8KBと16KBの両者が存在することが許可されている。このため、2048ブロック=4セグメント(1ブロック=8KB)のものと、1024ブロック=2セグメント(1ブロック=16KB)のものと2種類が在ることになる。1ブロック=16KBの場合には、1ブロック内のページ数は32ページとなる。

【0049】また、32MB、64MB、128MBの容量では、1ブロックあたりの容量は16KBのみであるとして規定される。従って、32MBでは2048ブロック=4セグメントとなり、64MBでは4096ブロック=8セグメントとなり、128MBでは8192ブロック=16セグメントとなる。

【0050】4-5 物理アドレス及び論理アドレスの概念

次に、上述したようなフラッシュメモリの物理的データ構造を踏まえたうえで、図10に示すデータ書き換え動作に従って、本例のファイルシステムにおける物理アドレスと論理アドレスの概念について説明する。

【0051】図10(a)には、或るセグメント内から4つのブロックを抜き出して、これを模式的に示している。各ブロックに対しては物理アドレスが付される。この物理アドレスはメモリにおけるブロックの物理的な配列順に従って決まるもので、或るブロックとこれに対応付けされた物理アドレスとの関係は不変となる。ここでは、図10(a)に示す4ブロックに対して、上から順に物理アドレスの値として、105、106、107、108が付されている。なお、実際の物理アドレスは2バイトにより表現される。

【0052】ここで、図10(a)に示すように、物理アドレス105、106で示されるブロックがデータの記憶されている使用ブロックで、物理アドレス107、108で示されるブロックがデータが消去(即ち、未記録領域)された未使用ブロックとなっている状態であるとする。

【0053】そして、論理アドレスであるが、この論理

アドレスは、ブロックに対して書き込まれたデータに付随するようにして割り振られるアドレスとされる。そして、この論理アドレスが、後述するFATファイルシステムが利用するアドレスとされている。図10(a)では、4つの各ブロックに対して、上から順に論理アドレスの値として、102, 103, 104, 105が付されている状態が示されている。なお、論理アドレスも実際には2バイトにより表現されるものである。

【0054】ここで、上記図10(a)に示す状態から、例えば物理アドレス105に格納されているデータの更新として、内容の書き換え又は一部消去を行うとする。このような場合、フラッシュメモリのファイルシステムでは、同じブロックに対して更新したデータを再度書き込むことはせずに、未使用のブロックに対してその更新したデータを書き込むようにされる。つまり、例えば図10(b)に示すようにして、物理アドレス105のデータは消去したうえで、更新されたデータはこれまで未使用ブロックであった物理アドレス107で示されるブロックに書き込むようにされる(処理①)。

【0055】そして、処理②として示すように、データ更新前(図10(a))の状態では物理アドレス105に対応していた論理アドレス102が、更新されたデータが書き込まれたブロックの物理アドレス107に対応するように、論理アドレスについての変更を行うものである。これに伴って、データ更新前は物理アドレス107に対応していた論理アドレス104については、物理アドレス105に対応するように変更されている。

【0056】つまり、物理アドレスはブロックに対して固有に付されるアドレスであり、論理アドレスは、一旦ブロックに対して書き込まれたデータに付随するようにしてついて回る、ブロック単位の書き込みデータに固有となるアドレスであるとみることができる。

【0057】このようなブロックのスワップ処理が行われることで、或る同一の記憶領域(ブロック)に対して繰り返し集中的にアクセスされることが無くなり、書き換え回数の上限があるフラッシュメモリの寿命を延ばすことが可能となる。そして、この際に論理アドレスを上記処理②のようにして扱うことで、ブロックのスワップ処理によって更新前と更新後のデータとで書き込まれるブロックの移動があるようにされても、FATからは同一のアドレスが見えることになり、以降のアクセスを適正に実行することができるものである。なお、後述する論理-物理アドレス変換テーブル上での更新のための管理を簡略にすることなどを目的として、ブロックのスワップ処理は、1セグメント内で完結するものとして規定されている。逆に言えば、ブロックのスワップ処理はセグメント間で跨るようにしては行われない。

【0058】4-6 論理-物理アドレス変換テーブル  
上記図10による説明から分かるように、ブロックのスワップ処理が行われることで、物理アドレスと論理アド

レスの対応は変化する。従って、フラッシュメモリに対するデータの書き込み及び読み出しのためのアクセスを実現するには、物理アドレスと論理アドレスとの対応が示される論理-物理アドレス変換テーブルが必要となる。つまり、論理-物理アドレス変換テーブルをFATが参照することで、FATが指定した論理アドレスに対応する物理アドレスが特定され、この特定された物理アドレスにより示されるブロックにアクセスすることが可能になるものである。逆に言えば、論理-物理アドレス変換テーブルが無ければ、FATによるフラッシュメモリへのアクセスが不可能となる。

【0059】従来では、例えばセット本体に対してメモリカード70が装着されたときに、セット本体側のマイクロプロセッサがメモリカード70の記憶内容をチェックすることで、セット本体側で論理-物理アドレス変換テーブルの構築を行い、更にこの構築された論理-物理アドレス変換テーブルをセット本体側のRAMに格納するようにしていた。つまり、メモリカード70内には、論理-物理アドレス変換テーブルの情報は格納されてはいなかった。これに対して本例では、以降説明するようにメモリカード70に対して、論理-物理アドレス変換テーブルを格納するように構成している。

【0060】図11は、本例のメモリカード70に対して格納される論理-物理アドレス変換テーブルの構築形態を概念的に示すものである。つまり、本例では、例えば論理アドレスの昇順に従って、これに対応する2バイトの物理アドレスを格納するようにしたテーブル情報を論理-物理アドレス変換テーブルとして構築するようにされる。なお、前述したように、物理アドレス、及び論理アドレスは共に2バイトで表現される。これは、128MBの最大容量のフラッシュメモリの場合には8192個のブロックが存在するため、最大で、この8192個のブロック数をカバーできるだけのビット数が必要とされることに基づく。このため、図11において例示している物理アドレスと論理アドレスとについても、実際に即して2バイトで表現している。但し、ここでは、この2バイトを16進数により表記している。つまり、「0x」によりその後続く値が16進法表記であることが示される。なお、この「0x」により16進数であることを表す表記は、以降の説明において16進数を表記する場合にも同様に用いることとする。(但し、表記の煩雑化を防ぐため「0x」を省略している図面もある。)

【0061】図12に、上記図11に示した概念に基づく論理-物理アドレス変換テーブルの構造例を示す。論理-物理アドレス変換テーブルは、フラッシュメモリの最後のセグメント内の或るブロックに対して、図12に示すようにして格納される。先ず図12(a)に示すように、ブロックを分割するページのうち、ページ0, 1からなる2ページの領域がセグメント0用の論理-物理

アドレス変換テーブルとして割り当てられる。例えば、図13にて説明したように、フラッシュメモリが4MBの容量であれば1セグメントしか有さないために、このページ0, 1のみの領域が論理-物理アドレス変換テーブルの領域となる。また、例えばフラッシュメモリが8MBの容量であれば2セグメントを有するため、セグメント0用の論理-物理アドレス変換テーブルとして割り当てられるページ0, 1に加え、これに続くページ2, 3の2ページがセグメント1用の論理-物理アドレス変換テーブルとして割り当てられることになる。

【0062】以降、フラッシュメモリの容量の増加に応じて、続く2ページごとにセグメントごとの論理-物理アドレス変換テーブルの割り当て領域が設定されていくことになる。そして、最大の128MBの容量を有する場合であれば16セグメントが存在するため、最大では、セグメント15用までのページが論理-物理アドレス変換テーブルの領域として割り当てられることになる。従って、最大の128MBの容量のフラッシュメモリでは、30ページが使用されることになり、図12(a)に示すページNとしては、最大でN=29となる。これまでの説明から分かるように、論理-物理アドレス変換テーブルは、セグメントごとに管理されるものである。

【0063】図12(b)は、1セグメントあたりの論理-物理アドレス変換テーブルの構造を示すものとして、2ページ分のデータエリアを抜き出して示している。つまり、1ページのデータエリアは512バイト(図8(e)参照)であることから、図12(b)には、1024(=512×2)バイトが展開されている状態が示されている。

【0064】図12(b)に示すように、この2ページ分のデータエリアである1024バイトについて2バイトごとに区切り、この2バイトごとの領域を、先頭から順次、論理アドレス0用、論理アドレス1用・・・、のようにして割付を行い、最後は先頭から991バイト目と992バイト目の2バイトの領域を論理アドレス495用の領域として割り付けるように規定を行う。これら2バイトごとの領域に対して、各論理アドレスが対応する物理アドレスを書き込むようにする。従って、本例の論理-物理アドレス変換テーブルでは、実際のデータ更新によるブロックのスワップ処理などにより物理アドレスと論理アドレスの対応が変更された場合には、論理アドレスを基準として、物理アドレスの格納状態が更新されるようにしてテーブル情報の書き換えが行われることになる。

【0065】また、残る993バイト目から最後の1024バイト目までの計32バイトの領域は、余剰ブロックの物理アドレスが格納される領域として割り当てられる。つまり、16個の余剰ブロックの物理アドレスを管理することができる。ここでいう余剰ブロックとは、例

えばブロック単位でデータの更新を行う際に書き換え対象となるデータを一時待避させる領域として設定されたいわゆるワークブロックなどを言うものである。

【0066】ところで、1セグメントは512ブロックに分割されているものであると先に説明したのにも関わらず、図12に示したテーブル構造では、管理可能なブロック数が論理アドレス0用～論理アドレス495用の496ブロックとしている。これは、實際上、上記した余剰アドレスが設定されることと、前述したように、フラッシュメモリでは、或ブロック数のディフェクト(使用不可領域)が許可されている。そのため現実には、相当数のディフェクトブロックが存在することに依る。従って、実際には、書き込み/消去が有効なブロックを管理するのに、496ブロックを管理できるように構成しておけば充分とされるものである。

【0067】そして、このようにして論理-物理アドレス変換テーブルが格納されるブロックについては、これを形成する各ページの冗長部における管理フラグ(図9参照)のデータ内容として、この管理フラグのビット3に対して‘0’がセットされることになる。これにより、当該ブロックが論理-物理アドレス変換テーブルが格納されているブロックであることが示されることになる。

【0068】論理-物理アドレス変換テーブルが格納されるブロックも、論理-物理アドレス変換テーブルの内容の書き換えがあった場合には、例外なく、先に図10にて説明したスワップ処理が行われる。従って、論理-物理アドレス変換テーブルが記録されているブロックは不定であり、或る特定のブロックに論理-物理アドレス変換テーブルを格納するように規定することは出来ない。そこで、FATは、フラッシュメモリにアクセスして上記した管理フラグのビット3が‘0’とされているブロックを検索することで、論理-物理アドレス変換テーブルが格納されているブロックを識別するようにされる。但し、論理-物理アドレス変換テーブルが格納されているブロックの検索がFATによって容易に行われるようにすることを考慮して、論理-物理アドレス変換テーブルが格納されているブロックは、そのフラッシュメモリ内における最後のナンバが付されたセグメントに在るように、本例では規定するものとされる。これにより、FATは最後のナンバが付されたセグメントのブロックのサーチだけで、論理-物理アドレス変換テーブルを検索することができる。つまり、論理-物理アドレス変換テーブルを検索するのに、フラッシュメモリの全てのセグメントを検索する必要は無いようにされる。上記図12に示した論理-物理アドレス変換テーブルは、例えばメモリカード70の製造時において格納するようにされる。

【0069】ここで、再度図13を参照して、フラッシュメモリ容量と論理-物理アドレス変換テーブルのサイ

ズとの関係を説明しておく。上記図11にて説明したように、1セグメントを管理するための論理-物理アドレス変換テーブルのサイズは2ページ分の1024バイト、つまり1KBとなる。従って、図13の最右列に記されているように、フラッシュメモリが4MB（1セグメント）の容量では論理-物理アドレス変換テーブルは1KBのサイズとなる。また、フラッシュメモリの容量が8MB（2セグメント）では論理-物理アドレス変換テーブルは2KB（4ページ）となる。また、フラッシュメモリの容量が16MBの場合、2048ブロック=4セグメントのものでは論理-物理アドレス変換テーブルは4KB（8ページ）、1024ブロック=2セグメントのものでは論理-物理アドレス変換テーブルは2KB（4ページ）となる。そして、フラッシュメモリの容量が32MB（4セグメント）では論理-物理アドレス変換テーブルは4KB（8ページ）、フラッシュメモリの容量が64MB（8セグメント）では論理-物理アドレス変換テーブルは8KB（16ページ）となり、フラッシュメモリの容量が最大の128MB（16セグメント）では論理-物理アドレス変換テーブルは16KB（32ページ）となる。

#### 【0070】4-7 ディレクトリ構造

メモ리카ード70に記録されるディレクトリ構成例を図14に示す。メモ리카ード70で扱うことのできる主データとしては、コンピュータ用データ、動画データ、静止画データ、メッセージデータ、オーディオデータ、制御用データなどがあるが、このためディレクトリ構造としては、ルートディレクトリから、「VOICE」（メッセージ用ディレクトリ）、「DCIM」（静止画用ディレクトリ）、「MOxxxxnn」（動画用ディレクトリ）、「CONTROL」（制御用ディレクトリ）、「HIFI」（オーディオ用ディレクトリ）、「PM」（情報処理装置用ディレクトリ）が配される。

【0071】そして図示していないが、各ディレクトリの下には、サブディレクトリやファイル（上述したデータベース）、フォルダ等が配され、いわゆるツリー構造の形態をとることになる。なお、もちろんこのようなディレクトリ構成は一例にすぎず、実際には情報処理装置1等による記録状況や記録されるファイル種別などに応じてディレクトリ構造が形成される。

#### 【0072】5. FAT構造

図7のファイルシステム階層で説明したように、ファイル管理処理はFATにより行われることになる。即ち図2に示した構成の情報処理装置1により、メモ리카ード70に対する記録再生（データ書込/読出）を実現するには、アプリケーション処理での要求に伴ってFATによるファイル記憶位置管理が参照され、さらに上述した論理-物理アドレス変換が行われて実際のアクセスが行われることになる。ここで、FATの構造について説明しておく。

【0073】図15はFATによる管理構造の概要を示している。なお、本例ではFAT及び論理-物理アドレス変換テーブルはメモ리카ード70内に格納されることになるが、図15に示すFAT構造が、メモ리카ード70内での管理構造となるものである。

【0074】図示するようにFAT管理構造は、パーティションテーブル、空き領域、ブートセクタ、FAT、FATのコピー、ルートディレクトリ、データ領域から成る。データ領域には、クラスタ2、クラスタ3・・・として単位データを示しているが、このクラスタとは、管理単位となるFATで扱う1データ単位である。一般にFATでは、クラスタサイズは標準で4Kバイトとされるが、このクラスタサイズは512バイト～32Kバイトの間で2のべき乗の大きさをとることができる。本例のメモ리카ード70では、上述したように1つのブロックが8Kバイト又は16Kバイトとされるが、1ブロック=8Kバイトとされるメモ리카ード70の場合は、FATで扱うクラスタは8Kバイトとされる。また1ブロック=16Kバイトとされるメモ리카ード70の場合は、FATで扱うクラスタは16Kバイトとされる。即ち、8Kバイト又は16KバイトがFAT管理上でのデータ単位であり、かつメモ리카ード70でのブロックとしての1つのデータ単位とされる。なお、従ってメモ리카ードからみれば、FATで扱われるクラスタサイズ=そのメモ리카ードのブロックサイズとなる。このため、本例の以降の説明については、簡略化のためにブロック=クラスタとして考えることとする。

【0075】そして図15左側にブロックナンバとして $x \cdots (x+m-1)$ 、 $(x+m)(x+m+1) \cdots (x+m+2) \cdots$ と示したが、例えばこのように各ブロックにおいてFAT構造を構築する各種データは記憶されることになる。なお、実際には必ずしもこのように物理的に連続する各ブロックに各情報が記憶されるものではない。

【0076】FAT構造において、まずパーティションテーブルには、FATパーティション（最大2Gバイト）の先頭と終端のアドレスが記述されている。ブート領域には、いわゆる12bitFAT、16bitFATの別や、FAT構造（大きさ、クラスタサイズ、各領域のサイズなど）が記述される。

【0077】FATは、後述するように各ファイルを構成するクラスタのリンク構造を示すテーブルとなり、またFATについては続く領域にコピーが記述される。ルートディレクトリには、ファイル名、先頭クラスタ番号、各種属性が記述される。これらの記述は1つのファイルにつき32バイト使用される。

【0078】FATにおいては、FATのエントリとクラスタは1対1で対応しており、各クラスタのエントリにはリンク先、つまり後に続くクラスタの番号が記述される。つまり、複数のクラスタ（=ブロック）で形成さ

れている或るファイルについてみると、まずディレクトリによって先頭のクラスタ番号が示され、FATにおけるその先頭クラスタのエントリには、次のクラスタ番号が示される。さらに次のクラスタ番号のエントリには、さらに次のクラスタ番号が示される。このようにクラスタのリンクがFATに記述される。

【0079】図16はこのようなリンクの概念を模式的に示している（数値は16進値）。例えば2つのファイル「MAIN. C」「FUNC. C」が存在するとすると、ディレクトリにはこの2つのファイルの先頭クラスタ番号が例えば「002」「004」と記述される。そしてファイル「MAIN. C」については、クラスタ番号「002」のエントリに次のクラスタ番号「003」が記述され、またクラスタ番号「003」のエントリに次のクラスタ番号「006」が記述される。さらに、クラスタ番号006がこのファイル「MAIN. C」の最後のクラスタであるとなると、クラスタ番号「006」のエントリには、最後のクラスタであることを示す「FFF」が記述される。これによりファイル「MAIN. C」がクラスタ「002」→「003」→「006」という順番で記憶されている。即ち、仮にクラスタ番号とメモリカード70でのブロック番号が一致していると仮定すると、ファイル「MAIN. C」は、メモリカード70内でブロック「002」「003」「006」に記憶されていることが表現されている。（但し、FATで扱うクラスタは、上述のように論理アドレスで扱うものとなるため、ブロックの物理アドレスとそのまま一致するものではない）

【0080】また同様にファイル「FUNC. C」については、FATにより、クラスタ「004」→「005」に記憶されていることが表現される。

【0081】なお、未使用のブロックに対応するクラスタについては、そのエントリは「000」とされる。

【0082】ところでルートディレクトリの領域に記憶される各ファイルのディレクトリにおいては、図16に示した先頭クラスタ番号だけでなく、例えば図17のように各種データが記述される。即ちファイル名、拡張子、属性、変更時刻情報、変更日付情報、先頭クラスタ番号、ファイルサイズが、それぞれ図示するバイト数で記述される。

【0083】また或るディレクトリの下層となるサブディレクトリについては、図15のルートディレクトリの領域ではなく、データ領域に記憶される。つまりサブディレクトリは、ディレクトリ構造を持つファイルとして扱われる。そしてサブディレクトリの場合はサイズは無制限とされ、また自分自身へのエントリと親ディレクトリへのエントリが必要になる。

【0084】図18に、或るルートディレクトリ内にファイル「DIR1」（属性＝ディレクトリ：つまりサブディレクトリ）があり、さらにその中にファイル「DI

R2」（属性＝ディレクトリ：つまりサブディレクトリ）があり、さらにその中にファイル「FILE」が存在する場合の構造例を示している。つまりルートディレクトリの領域には、サブディレクトリであるファイル「DIR1」としての先頭クラスタ番号が示され、上述したFATにより、クラスタX、Y、Zがリンクされている状態となる。この図からわかるように、サブディレクトリ「DIR1」「DIR2」についてはファイルとして扱われてFATのリンクに組み込まれる。

【0085】6. メモリカードと情報処理装置のインターフェース

図19により、メモリカード70と情報処理装置1のメモリカードインターフェース28の間のシリアルインターフェースシステム構成を説明する。メモリカード70内のコントロールIC80は、図19に示すようにフラッシュメモリコントローラ80a、レジスタ80b、ページバッファ80c、シリアルインターフェース80dとしての各ブロックを有するものとなっている。

【0086】フラッシュメモリコントローラ80aは、レジスタ80bに設定されたパラメータに基づいて、フラッシュメモリ81とページバッファ80cの間でのデータ転送を行う。そしてページバッファ80cにバッファリングされたデータはシリアルインターフェース80dを介して情報処理装置1のメモリカードインターフェース28側に転送され、また情報処理装置1のメモリカードインターフェース28から転送されてきたデータはシリアルインターフェース80dを介してページバッファ10cにバッファリングされる。

【0087】メモリカードインターフェース28側では、メモリカード70に対するインターフェース構造として、ファイルマネージャ60、転送プロトコルインターフェース61、シリアルインターフェース62を有する。ファイルマネージャ60はメモリカード70のファイル管理を行う。例えば本例のシステムではメモリカード70内にはメインデータファイルの管理のための管理ファイルが記憶されているが、情報処理装置1は装填されたメモリカード70から管理ファイルをよみこんでCPU22がファイルマネージャ60を形成することになる。メモリカード70へのアクセスはファイルマネージャ60に従って実行される。転送プロトコルインターフェース61は、レジスタ80b、ページバッファ80cへのアクセスを実行する。シリアルインターフェース62は、メモリカード70との間の3つの信号線、即ちSCLK（シリアルクロック）、BS（バーステイト）、SDIO（シリアルデータ入出力）において、任意のデータ転送を行うためのプロトコルを規定する。

【0088】以上の構成における各部の動作により、情報処理装置1によるメモリカード70（フラッシュメモリ81）に対する読出アクセス／書込アクセスが実行される。

【 〇 〇 8 9 】 7 . メモリカードへのファイル記録時の処理

以下、情報処理装置 1 によりメモリカード 7 0 に対してファイル（データベース）を記録する際の処理について説明していく。ここでは一例として、或るアプリケーションソフトウェアを構成する複数のファイルをメモリカード 7 0 に記録する場合の例を挙げ、まずその処理例を説明し、以降、その処理例によって形成されるディレクトリ構造例を各種示していく。なお、アプリケーションソフトウェアを構成する複数のファイルとは、例えば 1 又は複数の実行ファイル（リソースデータベース）と、1 又は複数のデータファイル（データベースデータベース）のことである。

【 〇 〇 9 〇 】 図 2 〇 は CPU 2 2 によって実行される記録時の処理を示している。ステップ F 1 〇 1 として CPU 2 2 は、ユーザーから或るアプリケーションを構成する複数のファイルをメモリカード 7 0 に記録する指示があったことを検出すると、ステップ F 1 〇 2 として、その複数のファイルを 1 つのディレクトリの下ファイルとして記録するために、ディレクトリを発生させる。このとき、ディレクトリの名称は、アプリケーション ID をそのまま用いる。アプリケーション ID とは、クリエーター ID と呼ばれるもので、システム上で各アプリケーションに対してそれぞれ固有の値として割り当てられている 3 2 バイトの ID コードであり、OS は、各アプリケーションをそのアプリケーション ID で識別して扱うものとなっている。

【 〇 〇 9 1 】 ところで、このアプリケーション ID は 3 2 バイトの数字列等であり、通常、ユーザーがその ID コードをみて、それによって示されるアプリケーションを判別できるものではない。そこで CPU 2 2 は、ステップ F 1 〇 3 で、アプリケーションをユーザーに提示する際に用いる名称（ユーザー提示用名称）を生成する。これは例えばアプリケーションソフトウェアの商品名称、機能名称などを利用して生成する。例えばアプリケーションソフトウェアとしてメモ帳ソフト、地図ソフト、予定表ソフトなどが存在する場合、これらのユーザー提示用名称として、「MEMO」「MAP」「SCHEDULE」等、ユーザーがすぐにアプリケーションを判別できるような名称を生成するものである。

【 〇 〇 9 2 】 続いてステップ F 1 〇 4 で、上記ディレクトリ名、即ちアプリケーション ID と、上記ユーザー提示用名称の対応情報を生成する。この対応情報としては、後述の例でそれぞれ述べるが、例えば対応テーブルファイルとされたり、ディレクトリとリンクされるファイルであったり、タグファイルであったり、ファイル内の情報であったりするなどの例が考えられる。いずれにしても、アプリケーション ID を名称とするディレクトリについて、ユーザー提示用名称を対応させるための情報である。

【 〇 〇 9 3 】 そしてステップ F 1 〇 5 では、実際にメモリカード 7 0 への記録処理を実行する。即ちステップ F 1 〇 2 で発生させたディレクトリ下に、アプリケーションを構成する各ファイルが配される状態となるようにメモリカード 7 0 への記録及び管理構造の更新を実行する。このとき上記対応情報も何らかの形で記録されるようにする。

【 〇 〇 9 4 】 本例の情報処理装置 1 では、このように記録が行われることで、システム及びユーザーの双方にとって好適なものとなる。即ち、或るアプリケーションソフトウェアから発生される 1 又は複数のファイルについて、1 つのディレクトリの下ファイルとして記録媒体に記録されることでファイル管理上好適であると共に、そのディレクトリの名称は、アプリケーションソフトウェアに固有に付されているアプリケーション ID を用いることから、当然に固有の名称としてディレクトリを設定できるものとなり、かつそのアプリケーションソフトウェアについてアプリケーション ID を用いる OS からすれば、ディレクトリ内容はその名称のみで明確に判別できるものとなる。従ってメモリカード 7 0 に記録したファイル管理上でも都合がよい。そしてさらに、ディレクトリ名称とユーザー提示用名称が対応づけられて記録されることで、ユーザーにディレクトリを提示する際には、ユーザーにとってわかりやすい名称とすることができ。例えばメモリカード 7 0 に記録されているファイルやディレクトリを表示部 2 に表示するような場合に、ディレクトリについてはユーザー提示用名称を用いることで、ユーザーはその名称からディレクトリの内容（ディレクトリ下に配されるファイルの種別）を判別できる。またそれによってユーザーのファイル指定、ディレクトリ指定等の操作性も向上する。即ちユーザーインターフェースが格段に向上する。

【 〇 〇 9 5 】 8 . 形成されるディレクトリ構造例 1  
上記処理例によっては、ステップ F 1 〇 4 で生成される対応情報によってディレクトリ名とユーザー提示用名称が対応づけられるが、その対応情報の形態例をとして 4 つの例を、それぞれ形成されるディレクトリ構造例 1 ～ 4 として説明していく。

【 〇 〇 9 6 】 まずディレクトリ構造例 1 を図 2 1 ( a ) に示す。これは対応情報が対応テーブルファイルとされる例である。なお、図 1 4 にも示したように、ルートディレクトリからは情報処理装置用ディレクトリ PM が形成され、例えばアプリケーションを構成するファイルは、このディレクトリ PM 下に記録されていくものとする。

【 〇 〇 9 7 】 図 2 1 の例は、例えばメモ帳ソフトとしてのアプリケーションについてファイル群が記録され、また地図ソフトとしてのアプリケーションについてファイル群が記録された場合を示している。まずメモ帳ソフトの構成ファイル群が記録される際には、例えば「 1 2 3

4・・・4321」という32バイトコードのアプリケーションIDが名称とされたディレクトリが形成され、そのディレクトリ下に、実行ファイル（xxx．prc等）やデータファイル（○○○．dtb等）が記録されることになる。そしてこのとき、ディレクトリPMの下に、上記ディレクトリ「1234・・・4321」と並列に対応テーブルファイルTBLが形成され、この対応テーブルファイルTBLには、図21（b）に示すようにディレクトリ「1234・・・4321」に対応して「MEMO」というユーザー提示用名称が示されるものとされる。

【0098】またその後、地図帳ソフトの構成ファイル群が記録される際には、例えば「9876・・・6789」という32バイトコードのアプリケーションIDが名称とされたディレクトリが形成され、そのディレクトリ下に、実行ファイル（abcd．prc）やデータファイル（aaa．dtb等）が記録されることになる。そしてこのとき、対応テーブルファイルTBLには、ディレクトリ「9876・・・6789」に対応して「MAP」というユーザー提示用名称が示される。

【0099】従ってシステムからみれば、ディレクトリPM下において、アプリケーションIDによって各ディレクトリの内容を判別できると共に、対応テーブルファイルによりディレクトリに対応するユーザー提示用名称を判別してユーザーに提示できる。

【0100】9．形成されるディレクトリ構造例2  
図22のディレクトリ構造例2は対応情報を、ディレクトリとリンクするファイルにより形成した例である。この場合、例えばメモ帳ソフトの構成ファイル群が記録される際には、ディレクトリPMの下に「1234・・・4321」という32バイトコードのアプリケーションIDが名称とされたディレクトリが形成され、そのディレクトリ下に、実行ファイル（xxx．prc等）やデータファイル（○○○．dtb等）が記録される。

【0101】そしてこのとき、ディレクトリPMの下に、上記ディレクトリ「1234・・・4321」と並列に、ユーザー提示用名称を示した名称ファイルMEMOが形成される。この名称ファイルは、ファイル名が「MEMO」とされファイル内容としては実体データの無いものとしてもよいし、実体データとして「MEMO」というユーザー提示用名称を記録したものでもよい。そして個の名称ファイル「MEMO」は、いわゆるシンボリックリンク、又はエイリアスと呼ばれるような方式で、ディレクトリ「1234・・・4321」にリンクされた状態とする。

【0102】従ってシステムからみれば、ディレクトリPM下において、アプリケーションIDによって各ディレクトリの内容を判別できると共に、リンクされる名称ファイルの存在によってディレクトリに対応するユーザー提示用名称を判別してユーザーに提示できる。

【0103】10．形成されるディレクトリ構造例3  
図23（a）のディレクトリ構造例3は対応情報を、ユーザー提示用名称を示したタグファイルとし、対応するディレクトリ下に配するようにした例である。この場合、例えばメモ帳ソフトの構成ファイル群が記録される際には、ディレクトリPMの下に「1234・・・4321」という32バイトコードのアプリケーションIDが名称とされたディレクトリが形成され、そのディレクトリ下に、実行ファイル（xxx．prc等）やデータファイル（○○○．dtb等）が記録される。

【0104】さらにこのとき、ディレクトリ「1234・・・4321」の下に、ユーザー提示用名称を示したタグファイルTGが形成される。このタグファイルTGは、図23（b）に示すように、ディレクトリ「1234・・・4321」に対応して「MEMO」というユーザー提示用名称が示されるファイルである。

【0105】従ってシステムからみれば、ディレクトリPM下において、アプリケーションIDによって各ディレクトリの内容を判別できると共に、タグファイルTGの存在によってディレクトリに対応するユーザー提示用名称を判別してユーザーに提示できる。

【0106】11．形成されるディレクトリ構造例4  
図24（a）のディレクトリ構造例4は対応情報を、ディレクトリ下の特定のファイルに記録されるユーザー提示用名称とする例である。この場合、例えばメモ帳ソフトの構成ファイル群が記録される際には、ディレクトリPMの下に「1234・・・4321」という32バイトコードのアプリケーションIDが名称とされたディレクトリが形成され、そのディレクトリ下に、実行ファイル（xxx．prc）やデータファイル（○○○．dtb等）が記録される。

【0107】さらにこのとき、実行ファイル（xxx．prc）には、図24（b）に示すように、「MEMO」というユーザー提示用名称と、ディレクトリ名「1234・・・4321」が記録される。

【0108】従ってシステムからみれば、ディレクトリPM下において、アプリケーションIDによって各ディレクトリの内容を判別できると共に、そのディレクトリ下のファイル群の特定のファイル、例えば実行ファイルからディレクトリに対応するユーザー提示用名称を判別してユーザーに提示できる。

【0109】以上、実施の形態としての情報処理装置の構成、記録媒体の例、記録時の処理、記録処理によるディレクトリ構造等を説明してきたが、本発明はこれらの例に限定されることなく、各種の変形例が考えられ、また本発明を適用できる装置は、多岐にわたるものである。

【0110】

【発明の効果】以上の説明から理解されるように本発明によれば、或るアプリケーションソフトウェアから発生

される 1 又は複数のファイルについて、1 つのディレクトリの下にファイルとして記録媒体に記録されることでファイル管理上好適であると共に、そのディレクトリの名称は、アプリケーションソフトウェアに固有に付されているアプリケーション ID を用いることから、容易に固有の名称としてディレクトリを設定でき、かつそのアプリケーションソフトウェアについてアプリケーション ID を用いるシステム上から、ディレクトリ内容は明確なものとする事ができる。これらのことからファイルシステム管理上で非常に都合がよい。そしてさらに、ディレクトリ名称とユーザ提示用名称が対応づけられて記録されることで、ユーザーにディレクトリを提示する際には、ユーザーにとってわかりやすい名称とすることができ、かつユーザーのディレクトリ名設定操作の負担もない。従ってシステム上のファイル管理性の向上とユーザーインターフェースの向上を同時に実現できるという効果がある。

【図面の簡単な説明】

【図 1】本発明の実施の形態の情報処理装置の平面図、右側面図、左側面図、上面図である。

【図 2】実施の形態の情報処理装置のブロック図である。

【図 3】実施の形態の情報処理装置の OS 構造の説明図である。

【図 4】実施の形態の情報処理装置で扱うデータベース構造の説明図である。

【図 5】実施の形態のメモリカードの外形形状を示す平面図、正面図、側面図、底面図である。

【図 6】実施の形態のメモリカードの内部構造の説明図である。

【図 7】実施の形態のファイルシステム処理階層の説明図である。

【図 8】実施の形態のメモリカードの物理的データ構造の説明図である。

【図 9】実施の形態のメモリカードの管理フラグの説明図である。

【図 10】実施の形態のメモリカードにおけるデータ更新処理と物理アドレス及び論理アドレスの概念の説明図である。

【図 11】実施の形態の論理-物理アドレス変換テーブルの管理形態の説明図である。

【図 12】実施の形態の論理-物理アドレス変換テーブルの構造の説明図である。

【図 13】実施の形態のメモリカードのフラッシュメモリ容量/ブロック数/1 ブロックの容量/1 ページの容量/論理-物理アドレス変換テーブルのサイズの関係の説明図である。

【図 14】実施の形態のメモリカードのディレクトリ構造の説明図である。

【図 15】FAT 構造の説明図である。

【図 16】FAT によるクラスタ管理形態の説明図である。

【図 17】ディレクトリの内容の説明図である。

【図 18】サブディレクトリ及びファイルの格納形態の説明図である。

【図 19】実施の形態の情報処理装置とメモリカードのインターフェース構成の説明図である。

【図 20】実施の形態の情報処理装置の記録時の処理のフローチャートである。

【図 21】実施の形態の形成されるディレクトリ構造例 1 の説明図である。

【図 22】実施の形態の形成されるディレクトリ構造例 2 の説明図である。

【図 23】実施の形態の形成されるディレクトリ構造例 3 の説明図である。

【図 24】実施の形態の形成されるディレクトリ構造例 4 の説明図である。

【符号の説明】

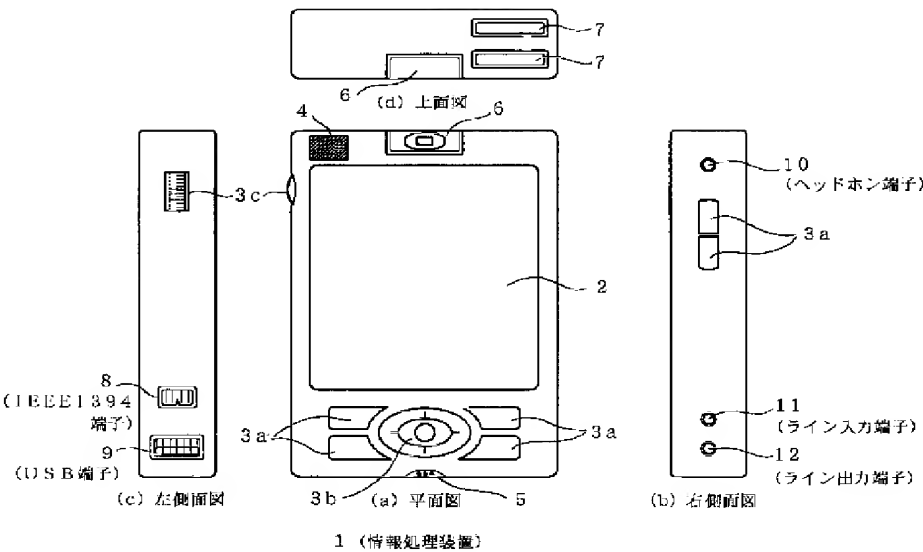
1 情報処理装置、2 表示部、3 a, 3 b, 3 c 操作子、4 スピーカ、5 マイクロホン、6 撮像部、7 メモリスロット、8 IEEE 1394 端子、9 USB 端子、10 ヘッドホン端子、11 ライン入力端子、12 ライン出力端子、21 システムコントローラ、22 CPU、23 フラッシュ ROM、24 DRAM、25 USB インターフェース、26 IEEE 1394 インターフェース、27 表示制御部、28 メモリカードインターフェース、29 オーディオインターフェース、70 メモリカード

【図 17】

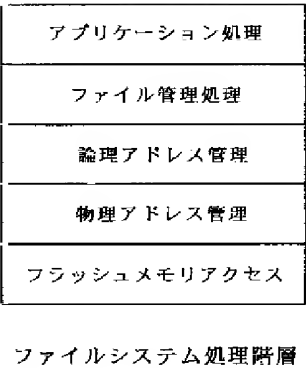
ファイル名 1 本分のディレクトリの構造。( ) はバイト数。

ファイル名 (8)	拡張子 (3)	属性 (1)	リザーブ (16)	時刻 (2)	日付 (2)	先頭クラスタ (2)	サイズ (4)
-----------	---------	--------	-----------	--------	--------	------------	---------

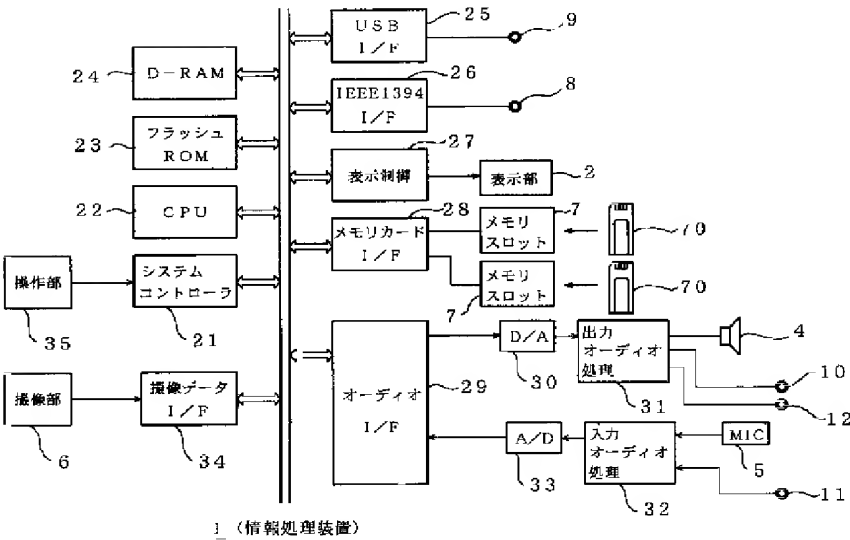
【図1】



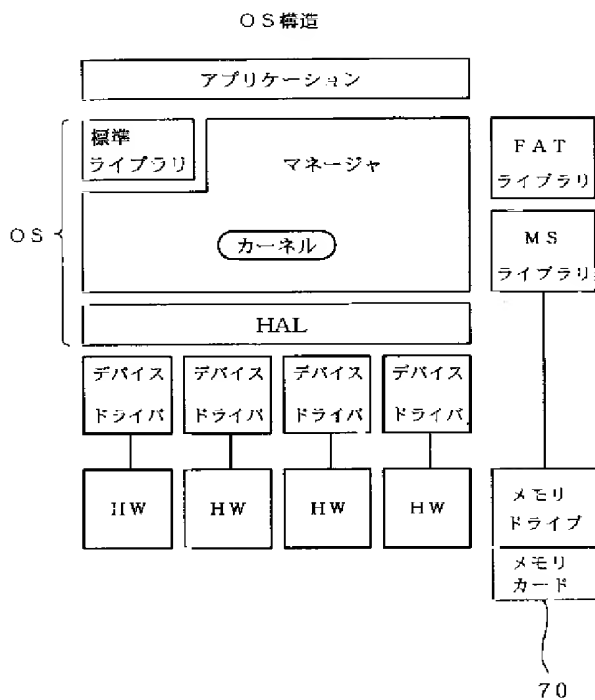
【図7】



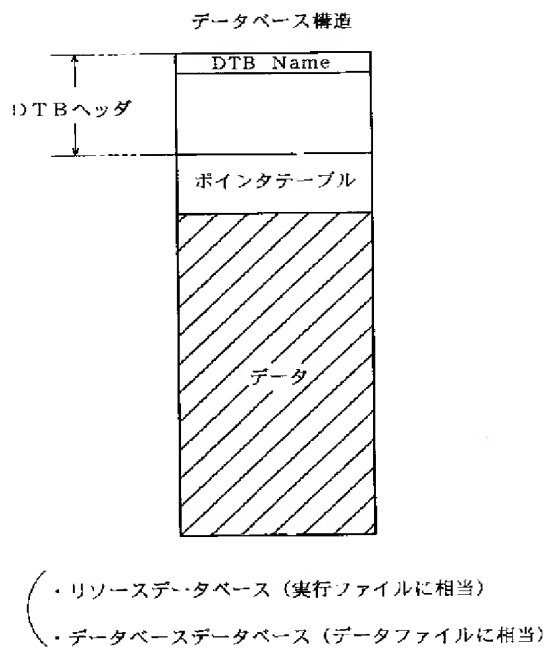
【図2】



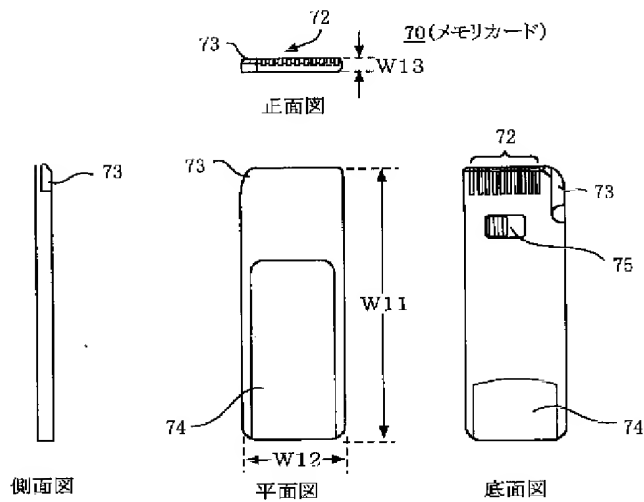
【図3】



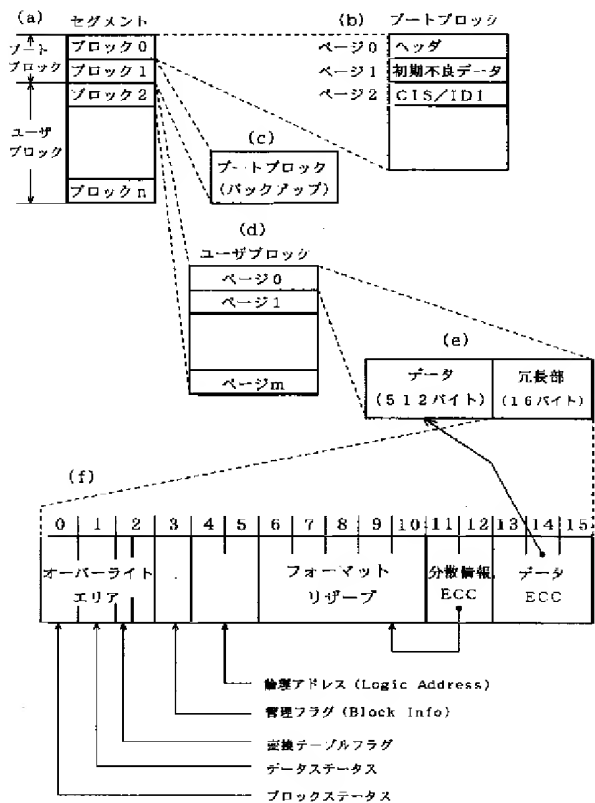
【図4】



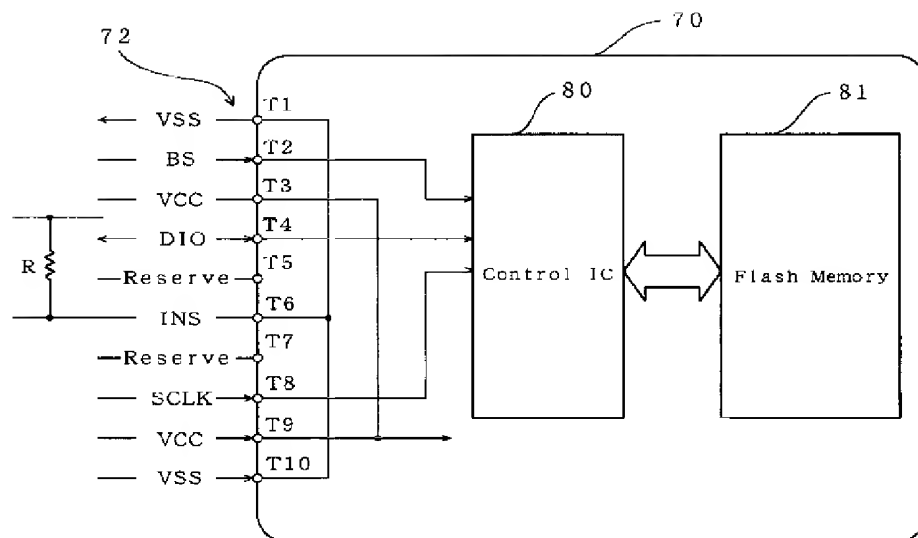
【図5】



【図8】



【図6】

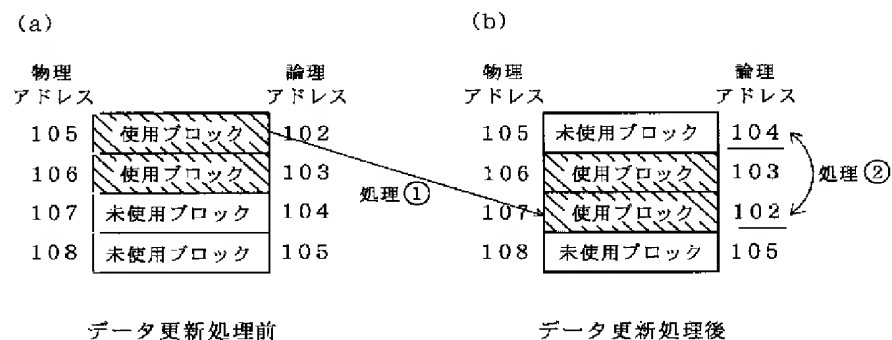


【例9】

## 管理フラグ

ビット	定義
7	リザーブ
6	リザーブ
5	アクセス許可 (1:free 0:Read Protected)
4	コピー禁止指定 (1:OK 0:NG)
3	変換テーブルフラグ (1:無効 0:テーブルブロック) *最終セグメントのみ有効
2	システムフラグ (1:ユーザブロック 0:ブートブロック)
1	リザーブ
0	リザーブ

【図 10】



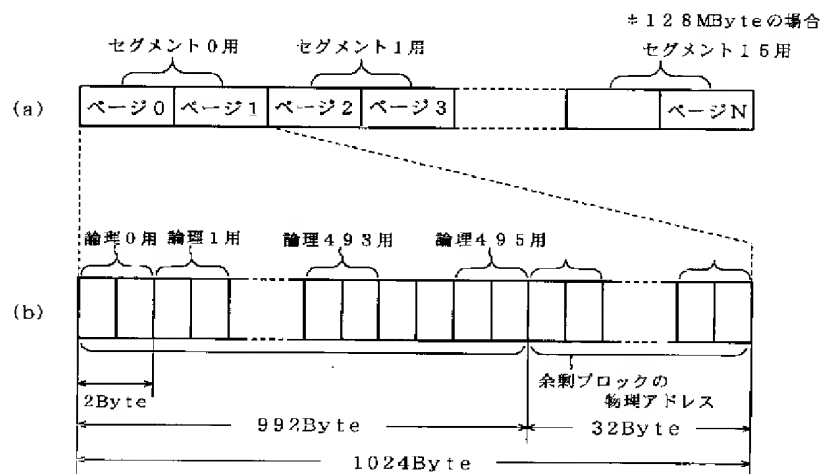
【図11】

物理アドレス (2Byte)		論理アドレス (2Byte)
0x00	0x03	←論理アドレス 0x0000
0x00	0x04	←論理アドレス 0x0001
0x00	0x04	←論理アドレス 0x0002
0x00	0x05	←論理アドレス 0x0003
0x01	0xA8	←論理アドレス 0x0004
0x00	0x06	←論理アドレス 0x0005

論理アドレス順に、対応する物理アドレスを格納する。

【図12】

論理／物理アドレス変換テーブル

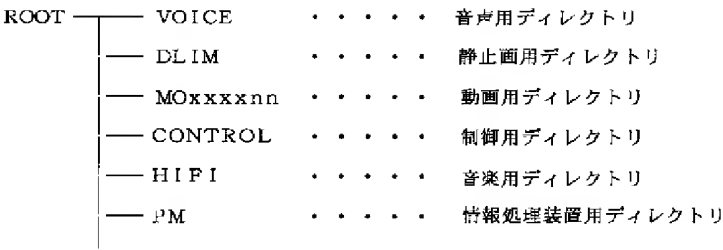


【図13】

フラッシュメモリ 容量	ブロック数	1ブロックの 容量	1ページの 容量	論理／物理アドレス 変換テーブルサイズ
4MB	512 (1セグメント)	8KB (16page)	(512+16) B	1KB (2page)
8MB	1024 (2セグメント)	8KB (16page)	(512+16) B	2KB (4page)
16MB	2048 (4セグメント)	8KB (16page)	(512+16) B	4KB (8page)
	1024 (2セグメント)	16KB (32page)	(512+16) B	2KB (4page)
32MB	2048 (4セグメント)	16KB (32page)	(512+16) B	4KB (8page)
64MB	4096 (8セグメント)	16KB (32page)	(512+16) B	8KB (16page)
128MB	8192 (16セグメント)	16KB (32page)	(512+16) B	16KB (32page)

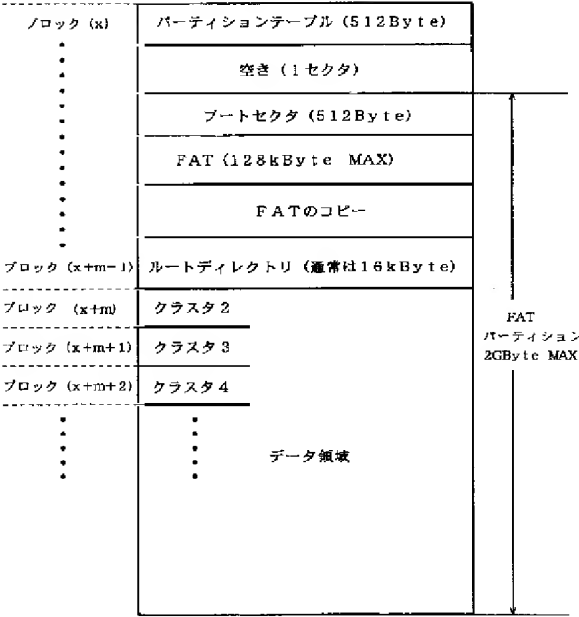
【図14】

ディレクトリの構成



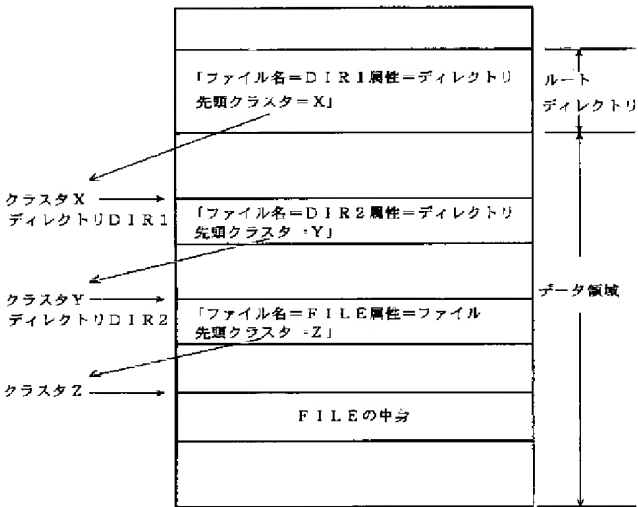
【図15】

FAT構造概要



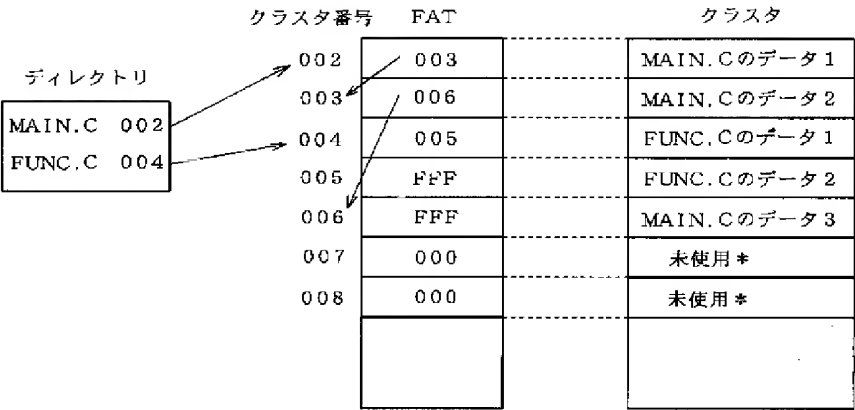
【図18】

ファイル≠DIR1≠DIR2≠FILEの配置

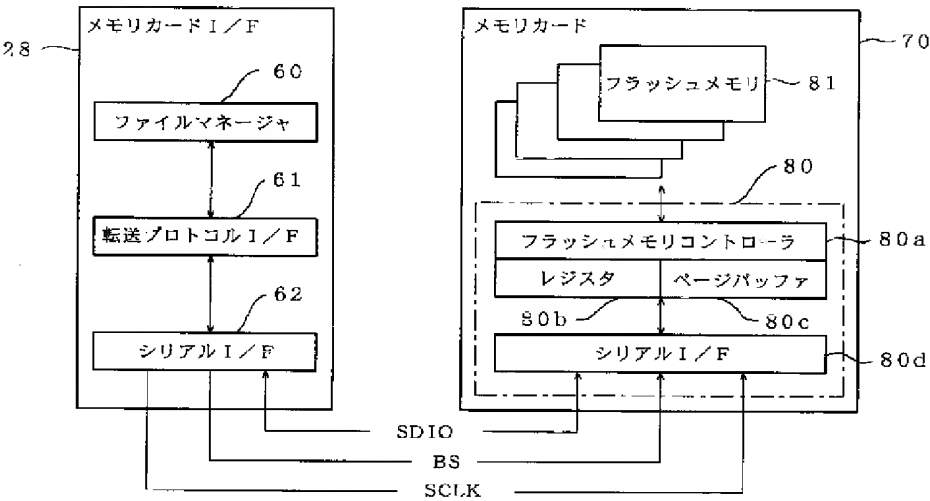


【図16】

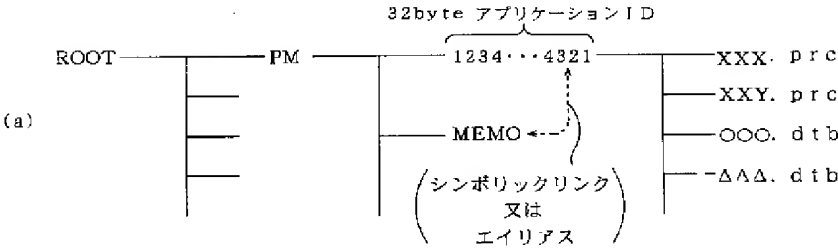
FAT概念図



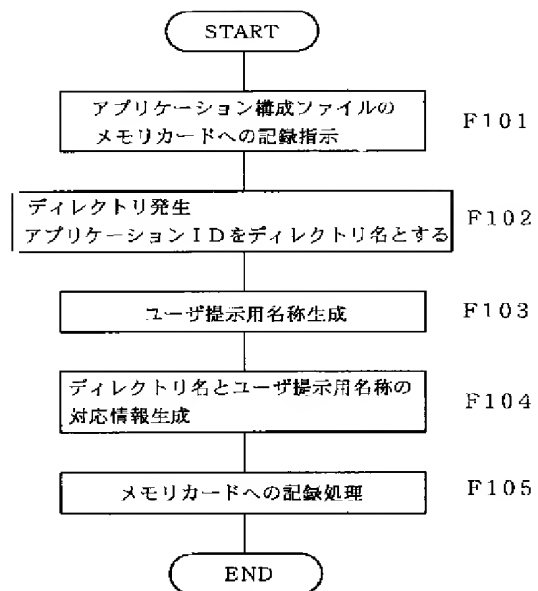
【図19】



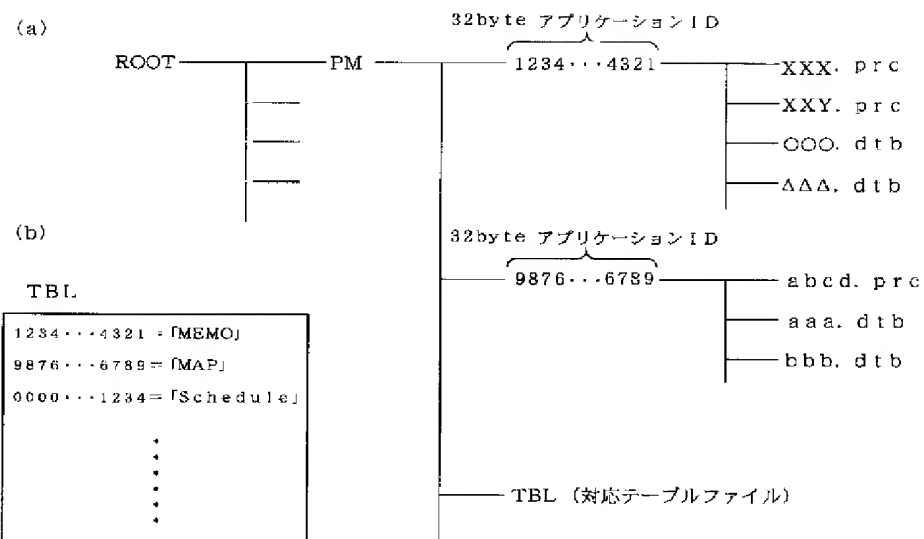
【図22】



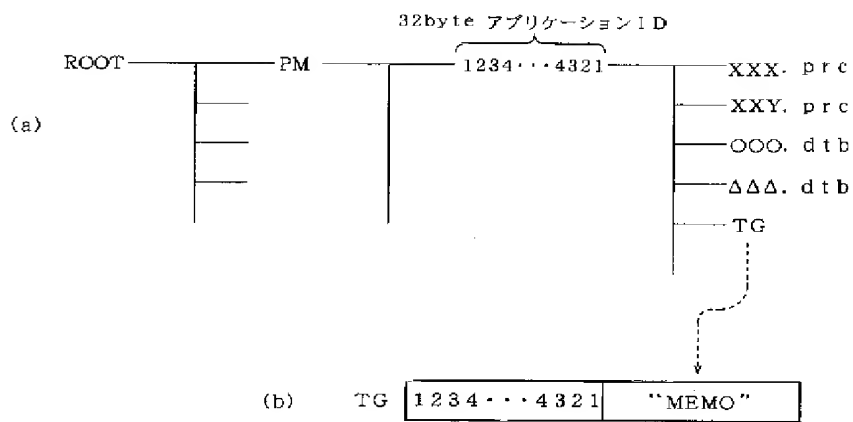
【图 20】



【例 2-1】



【 図 2 3 】



【 図 2 4 】

